LISTING OF CLAIMS:

1. (previously amended) A software object included in a computer system, comprising:

a platform dependent method; and

a platform independent wrapper arranged to call the platform dependent method, wherein a platform independent object accesses the platform dependent method by calling the wrapper, wherein the wrapper then calls the platform dependent method.

2. (original) A software object as recited in claim 1, wherein substantially the only operation performed by the wrapper is to act as an intermediary between the platform independent object and the native method to facilitate calling the platform dependent native method from the platform independent object.

3. (original) A software object as recited in claim 2, wherein the software object is one of a plurality of software objects included in the computer system.

4. (original) A software object as recited in claim 3, wherein the platform dependent method is one of a plurality of platform dependent methods.

5. (original) A software object as recited in claim 4, wherein the wrapper is one of a plurality of wrappers each being arranged to call an associated one of the plurality of platform dependent methods.

6. (original) A software object as recited in claim 5, wherein a first software object includes a first wrapper and an associated first method designed to run on a first platform.

7. (original) A software object as recited in claim 6, wherein a second software object includes a second wrapper and an associated second method designed to run on a second platform that is different than the first platform.

8. (original) A software object as recited in claim 7, wherein the wrapper is a Java wrapper.

9. (original) A software object as recited in claim 8, wherein the platform independent object is a Java device driver.

10. (previously amended) A computer-implemented method of accessing a platform dependent method by a platform independent object in a computer system, the computer system having an encapsulation object, the method comprising:

calling a platform independent wrapper by the platform independent object; and

calling the method by the wrapper wherein the wrapper is included in the encapsulation object.

11. (original) A computer implemented method as recited in claim 10, wherein substantially the only operation performed by the wrapper is to act as an intermediary between the platform independent object and the method so as to facilitate calling the platform dependent method from the platform independent object.

12. (original) A computer implemented method as recited in claim 10, wherein the encapsulation object is one of a plurality of encapsulation objects included in the computer system, and wherein the wrapper is one of a plurality of wrappers included in the computer system, and wherein the method is one of a plurality of methods included in the computer system.

13. (original) A computer implemented method as recited in claim 12, wherein the computer system includes a first encapsulation object containing a first wrapper and an associated first method, wherein the first method is designed to run on a first platform.

14. (original) A computer implemented method as recited in claim 13, wherein the computer system includes a second encapsulation object containing a second wrapper and an associated second method that is designed to run on a second platform that is different than the first platform.

15. (original) A computer implemented method as recited in claim 14, wherein the platform independent object accesses the first method by calling the first wrapper that, in turn, calls the first method.

16. (original) A computer implemented method as recited in claim 14, wherein the platform independent object accesses the second method by calling the second wrapper that, in turn, calls the second method.

17. (previously amended) A computer-implemented method of accessing a platform dependent method by a platform independent object in a computer system, the computer system including a system manager, an encapsulation object that contains the platform dependent method and a business card associated with the platform independent object, the business card containing configuration data that includes an encapsulation object pointer that is used to identify the encapsulation object containing the platform dependent method, comprising:

requesting the encapsulation object containing the platform dependent method from the system manager by the platform independent object;

retrieving the business card corresponding to the requesting object by the system manager;

retrieving the encapsulation object pointer from the requesting object's business card by the system manager;

instantiating the encapsulation object identified by the encapsulation object pointer;

calling a platform independent wrapper contained in the encapsulation object corresponding to the platform dependent method by the platform independent object; and

calling the platform dependent method by the wrapper.

18. (previously presented) The method as recited in claim 17, wherein the business card is instantiated by a system administrator at system startup.

19. (original) The method as recited in claim 18, wherein the platform independent object is a device driver, wherein the device driver is used to manage a device coupled to the computer system.

20. (original) The method as recited in claim 19, wherein the system manager is a bus manager used to manage a bus coupled to the device.

21. (original) A computer implemented method of transforming a mixed software object containing platform independent code and platform dependent code into a pure software object having only platform independent code, comprising:

instantiating an encapsulating object containing all the platform dependent code, the encapsulating object including a wrapper associated with and used to invoke the platform dependent code;

removing all the platform dependent code from the mixed software object; and

replacing all calls to the platform dependent code in the mixed software object with calls to the wrapper included in the native encapsulating object.

22. (original) A computer program product for transforming a mixed software object containing platform independent code and platform dependent code into a pure software object having platform independent code only, the computer program product comprising:

computer code that writes an encapsulating object containing all the platform dependent code, the encapsulating object including a Java wrapper used to invoke the platform dependent code;

computer code that removes all the platform dependent code from the mixed software object; and

computer code that replaces all calls to the platform dependent code in the mixed software object with calls to the wrapper included in the native encapsulating object.

23. (new) A native encapsulation object included in a computer system, the native encapsulation object comprising:

a plurality of device drivers; and

a plurality of wrappers arranged to call an associated device driver of the plurality of device drivers, wherein a platform independent object accesses an associated device driver by calling one of the plurality of wrappers, wherein the wrapper then calls the associated device driver.